BIOSTATS 640 – Intermediate Biostatistics
Spring 2022

Introduction to R
04 – Introduction to `tidyverse & dplyr` with Illustration of Simple Linear Regression

Introduction.

Your first R Markdown Data Analysis and Report! We now have sufficient foundations in place that we can begin to work with data and report our findings. Specifically, at this point, you should have some experience with: (1) importing an excel dataset (2) examining the structure of a dataset (3) producing descriptive statistics; and (4) producing a graph.

In this lesson, we will learn a little bit about the package **`tidyverse`**, in particular how to use one of its component packages, **`dplyr,`** for basic data manipulations. Nice! I will also encourage you to adopt a structured approach to data management, analysis and report writing using R Markdown.

The dataset we will use is the 1970 Draft Lottery dataset (**`draftlottery1970.xlsx`**). We will produce some descriptive statistics and visualizations and will perform a simple linear regression.

As needed, see again R lesssons 01 - 03:
01 – R Essentials (R Studio interface, the console as a giant calculator, a few basic commands)
02 – Introduction to Packages and Simple Data Description
03 - Working Directory, R Markdown and Data Inspection

Before you begin

__1. If you have not already done so, install the package **`tidyverse`**
Note: **`tidyverse`** is actually a bundle of packages, of which **`dplyr`** is one.
So you only need to do a one-time installation of **`tidyverse`**

__2. Download from the course website the dataset **draftlottery1970.xlsx** and place it in your working directory.

1. Introduction to the 1970 Draft Lottery Dataset

draftlottery1970.xlsx

The Vietnam War was a 20-year long (1955-1975) conflict between the government of North Vietnam and its Viet Cong allies in South Vietnam versus the government of South Vietnam and its principal ally, the United States. The United States sent in combat forces beginning in 1965. At this time in the United States, young men were required to register for military service and thus, possibly, be sent to Vietnam to fight. Military service, and in particular being sent to Vietnam, could sometimes be avoided but it was increasingly understood to be an unfair system.

In an effort to make the system more fair, a draft lottery was established. The first draft lottery took place on December 1, 1969. **It is known as the 1970 Draft Lottery.**

The 1970 Draft Lottery was highly controversial and was ultimately determined to be not random.

Source: *http://ww2.amstat.org/publications/jse/v5n2/datasets.starr.html#fienberg1*

"This lottery was a source of considerable discussion before being held on December 1, 1969. Soon afterwards a pattern of unfairness in the results led to further publicity: those with birthdates later in the year seemed to have had more than their share of low lottery numbers and hence were more likely to be drafted. On January 4, 1970, the *New York Times* ran a long article, "Statisticians Charge Draft Lottery Was Not Random," illustrated with a bar chart of the monthly averages (Rosenbaum 1970a). It described the way the lottery was carried out, and with hindsight one can see how the attempt at randomization broke down. The capsules were put in a box month by month, January through December, and subsequent mixing efforts were insufficient to overcome this sequencing. The details of the procedure are quoted in Fienberg (1971a) and the first three editions of Moore (1979, 1985, 1991)."

Data dictionary/Codebook

Position	Variable	Variable Label	Type	Codes	Missing data
1	day_birth	Day of year born	numeric	Range: [1, 366]	None
2	draft_number	Rank assignment	numeric	Range: [1, 366]	None
3	month_birth	Month of year born	numeric	Range: [1, 12] 1 = "January" 2 = "February" .... 12 = "December"	None

2. Highlights from Lesson 03
Working Directory, R Markdown, and Data Inspection

1. The **working directory** is a **path** or **folder** on your computer (or in the cloud, depending). It is R's "go to". This is the location where files will be read from and written to.
   To set: SESSION > SET WORKING DIRECTORY > CHOOSE DIRECTORY

2. Recommended use of 3 of the panes in R Studio
   Console: giant calculator, help
   R Script: "Journal" your commands and comments here (e.g., a nifty place to catalogue great R fragments)
   R Markdown: Produce an archivable record of your code + output + narrative in a pretty format (e.g., html, pdf)

3. R Markdown **text** cannot be "word processed" using MS Word. Instead we use a **text markup** language to do these tasks. More on this in a future R lesson!

4. R Markdown R **commands** are embedded in gray-shaded blocks called **chunks**

5. Producing your **final report** involves **knitting/rendering** to your desired output format (e.g., html, pdf, word and others)

3. Introduction to `tidyverse` and `dplyr`

Before you begin - Video
Consider watching this video.
(Source: R Programming 101) Manipulate data using the tidyverse: **`select`**, **`filter`**, and **`mutate`** (video, 6:55)

There are multiple ways to accomplish data cleaning/preparation. You could use functions that are in the {base} package that came pre-installed when you installed R. An alternative, and very nice, approach is to use functions that are in the {dplyr} package. The {dplyr} package is actually one of several packages that are contained within the package {tidyverse}.

`tidyverse` includes 8 core packages + 11 additional packages.

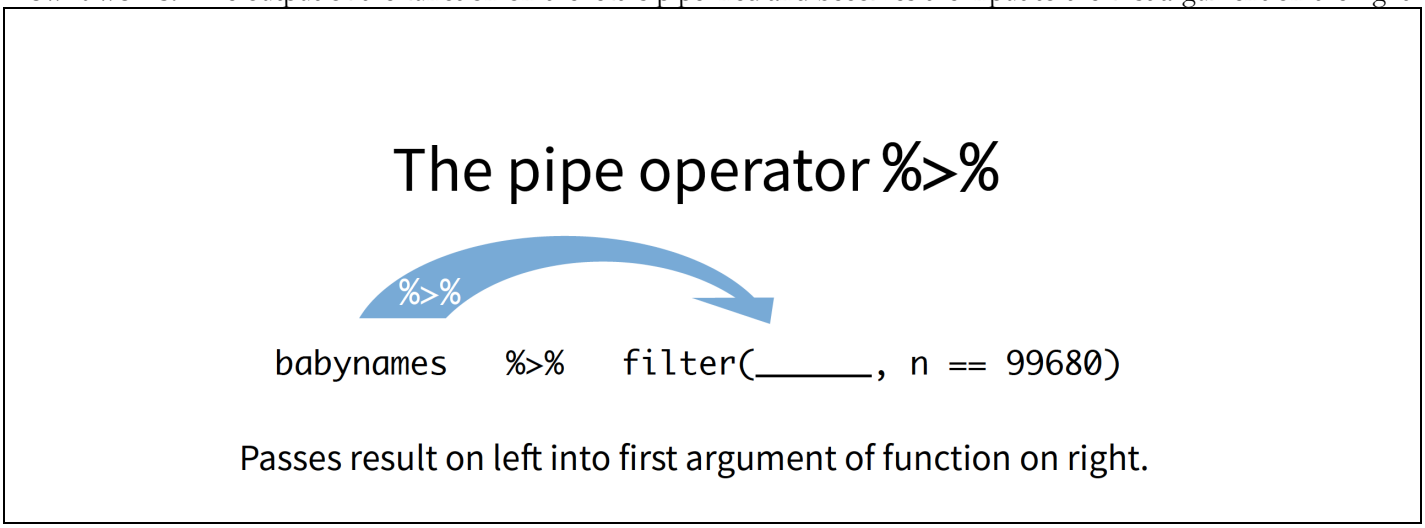
- `dplyr` is one of the core packages of tidyverse
- if you have issued `library(tidyverse)` you do NOT need to issue `library(dplyr)`

There are lots of things you can do in tidyverse. In this R lesson, we introduce some key fundamentals!

1. `%>%` How to use the pipe operator to "chain" a series of commands (you'll be glad)
2. `select( )` Select the variables you want to work with
2. `filter( )` Consider certain observations only
3. `mutate( )` Create a new variable

Introduction to the pipe operator `%>%`
Use this to "chain" a series of commands (you'll be glad)
In words: `%>%` is saying to R "**`AND THEN`**"

How it works. The output of the function on the left is pipelined and becomes the input to the first argument on the right.



Source: Garrett Grolemund, R Studio

Example

```
library(tidyverse)                                          # attach ALL core packages of {tidyverse}

iris %>%                                                 # start with dataframe iris.  AND THEN
    filter(Species=="setosa") %>%                           # filter observations.  AND THEN
    select(Sepal.Length) %>%                                # select variable.  AND THEN
    mutate(length2 = Sepal.Length^2)                        # mutate to create new variable length2
```

Good to Know

dplyr commands for **variables** (COLUMNS)

%>%	Pipe operator (translation: "And then")
select( )	Use to select variables (columns)
rename( )	Use to rename a variable (column)
mutate( )	Use to create new variable(column)
relocate( )	Use to rearrange the order of the columns

Good to Know

dplyr commands for **observations** (ROWS):

%>%	Pipe operator (translation: "And then")
slice( )	Use to select observations by their position (rows)
filter( )	Use to select observations if they meet criteria defined by variables (rows)
arrange( )	Use to sort the observations (rows)

4. Structure Your R Markdown Work

Structuring your R Markdown work has several advantages

- Clarity! You and others will be able to read your work later and know what you did.
- Structuring forces you to think through all the steps involved in your R work; and
- Error checking and correction is much easier

Guidelines for Structuring your R Markdown.
This is your choice, obviously. Design a structure that works best for you.

Here, I share my structure
- Ordered series of chunks
- Each chunk does just one task

```
__1.     Initialize Session

__2.     Set Working Directory

__3.     Import Raw Data

__4.     Create Data for This Session (Filter Observations, Select Variables)

__5.     Create New Variables

__6.     Descriptives

__7.     Analysis (typically multiple chunks)

__8.     Reporting:  Numerical (typically multiple chunks)

__9.     Reporting:  Graphs (typically multiple chunks)
```

5. Your First R Markdown Data Analysis and Report
Simple Linear Regression

Dear class, FYI - blue coloring of output is mine - cb.

initialize session

```
setwd("/cloud/project")                 # Set working directory
getwd()                                 # Check working directory
options(scipen=999)                     # Turn off scientific notation
rm(list = ls())                         # Clear the Decks
```

import raw data

```
library(readxl)
# paste in commands from FILE > IMPORT DATASET > FROM EXCEL here
# Don't forget: the command View( ) cannot be knitted, so do not paste it here
draftlottery1970 <- read_excel("draftlottery1970.xlsx")              # paste from code provided upon importing with menu
str(draftlottery1970)                                                # str( ) to examine structure of data

## tibble [366 × 3] (S3: tbl_df/tbl/data.frame)
##  $ day_birth   : num [1:366] 1 2 3 4 5 6 7 8 9 10 ...
##  $ draft_number: num [1:366] 305 159 251 215 101 224 306 199 194 325 ...
##  $ month_birth : num [1:366] 1 1 1 1 1 1 1 1 1 1 ...
```

filter observations and select raw variables of interest

```
# Be sure to have done a one-time installation of the package {tidyverse}

library(tidyverse)                          # package {tidyverse} must be attached to each session

junk <- draftlottery1970 %>%                # use dataframe draftlottery170. AND THEN
         filter(month_birth==6) %>%         # filter( ) to restrict observations to month_birth is JUNE. AND THEN
         select(day_birth,draft_number)     # select( ) to select variables of interest.

junk                                        # show
## # A tibble: 30 × 2
##    day_birth draft_number
##        <dbl>        <dbl>
##  1       153          249
##  2       154          228
##  3       155          301
##  4       156           20
##  5       157           28
##  6       158          110
##  7       159           85
##  8       160          366
##  9       161          335
## 10       162          206
## # … with 20 more rows
```

create new variables

```
# Be sure to have done a one-time installation of the package {tidyverse}
library(tidyverse)                          # package {tidyverse} must be attached to each session

junk <- junk %>%                            # use dataframe junk. AND THEN
        mutate(newvar1 = draft_number^2,    # mutate( ) to create new variables
               newvar2 = day_birth^2)       # Tip - Each on its own line

junk                                        # show
```

```
## # A tibble: 30 × 4
##    day_birth draft_number newvar1 newvar2
##        <dbl>        <dbl>   <dbl>   <dbl>
##  1       153          249   62001   23409
##  2       154          228   51984   23716
##  3       155          301   90601   24025
##  4       156           20     400   24336
##  5       157           28     784   24649
##  6       158          110   12100   24964
##  7       159           85    7225   25281
##  8       160          366  133956   25600
##  9       161          335  112225   25921
## 10       162          206   42436   26244
## # … with 20 more rows
```

create monthf as a factor version of the variable month_birth (REMEMBER not to overwrite raw variable)

```
draftlottery1970$monthf <- factor(draftlottery1970$month_birth)   # STEP 1: initialize new var as factor

draftlottery1970$monthf <- factor(draftlottery1970$monthf,        # STEP 2:  create value labels
                                  levels=c(1,2,3,4,5,6,7,8,9,10,11,12),
                                  labels=c("January","February", "March", "April",
                                           "May", "June", "July", "August",
                                           "September", "October", "November", "December"))
str(draftlottery1970)
## tibble [366 × 4] (S3: tbl_df/tbl/data.frame)
##  $ day_birth   : num [1:366] 1 2 3 4 5 6 7 8 9 10 ...
##  $ draft_number: num [1:366] 305 159 251 215 101 224 306 199 194 325 ...
##  $ month_birth : num [1:366] 1 1 1 1 1 1 1 1 1 1 ...
##  $ monthf      : Factor w/ 12 levels "January","February",..: 1 1 1 1 1 1 1 1 1 1 ...
```

descriptives of study cohort

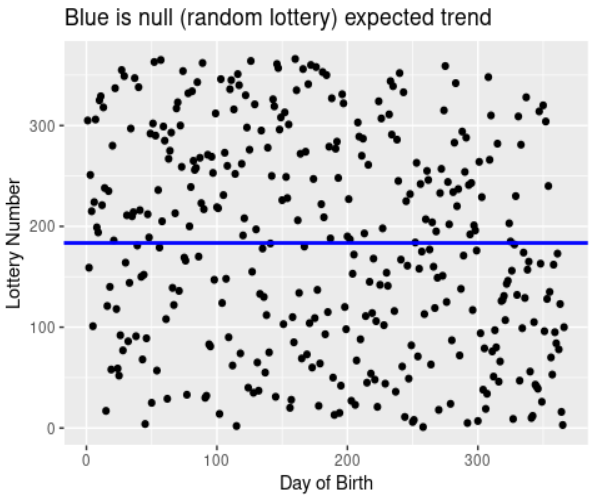
```
library(summarytools)                                                  # attach {summarytools} to access freq( )
library(stargazer)                                                     # attach {stargazer} to access stargazer( )

summary(draftlottery1970)                                              # summary( ) for no frills descriptives
##    day_birth       draft_number     month_birth         monthf
##  Min.   :  1.00   Min.   :  1.00   Min.   : 1.000   January: 31
##  1st Qu.: 92.25   1st Qu.: 92.25   1st Qu.: 4.000   March  : 31
##  Median :183.50   Median :183.50   Median : 7.000   May    : 31
##  Mean   :183.50   Mean   :183.50   Mean   : 6.514   July   : 31
##  3rd Qu.:274.75   3rd Qu.:274.75   3rd Qu.: 9.750   August : 31
##  Max.   :366.00   Max.   :366.00   Max.   :12.000   October: 31
##                                                     (Other):180

draftlottery1970 <- data.frame(draftlottery1970)           # stargazer( ) requires that data argument be a dataframe
stargazer(draftlottery1970,                                # stargazer( ) to obtain a nicer looking set of descriptives
          type="text",
          summary.stat=c("n", "min", "max"),               # option summary.stat=c( ) to choose statistics to report
          title="Draft Lottery 1970")                      # option title="STUFF" to obtain title

##
## Draft Lottery 1970
## ========================
## Statistic     N  Min Max
## ------------------------
## day_birth    366  1  366
## draft_number 366  1  366
## month_birth  366  1  12
## ------------------------
```

```
freq(draftlottery1970$monthf)                          # freq( ) for freq table of categorical var(factor in R)
## Frequencies
## draftlottery1970$monthf
## Type: Factor
##
##                  Freq   % Valid   % Valid Cum.   % Total   % Total Cum.
## --------------- ------ --------- -------------- --------- --------------
##         January     31      8.47           8.47      8.47           8.47
##        February     29      7.92          16.39      7.92          16.39
##           March     31      8.47          24.86      8.47          24.86
##           April     30      8.20          33.06      8.20          33.06
##             May     31      8.47          41.53      8.47          41.53
##            June     30      8.20          49.73      8.20          49.73
##            July     31      8.47          58.20      8.47          58.20
##          August     31      8.47          66.67      8.47          66.67
##       September     30      8.20          74.86      8.20          74.86
##         October     31      8.47          83.33      8.47          83.33
##        November     30      8.20          91.53      8.20          91.53
##        December     31      8.47         100.00      8.47         100.00
##            <NA>      0                                0.00         100.00
##           Total    366    100.00         100.00    100.00         100.00


print(dfSummary(draftlottery1970), method='render')      # Pretty but will not knit/render to MS WORD or pdf (depending)
```

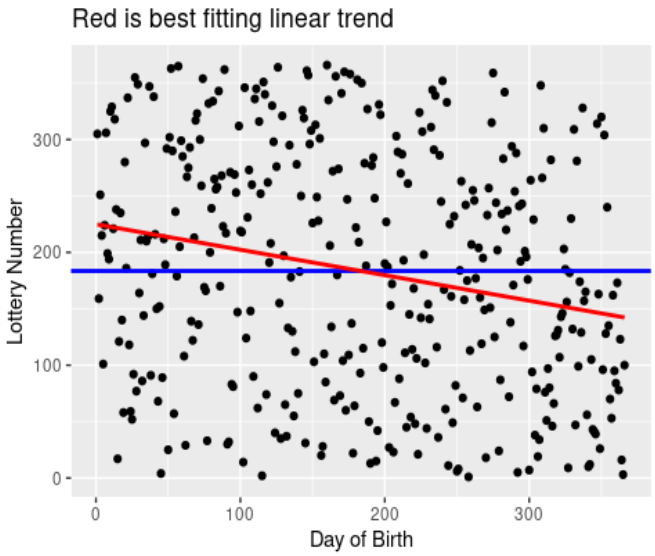
analysis - data visualizations w overlay null and best fitting lines

```
library(ggplot2)

# simple XY scatterplot
ggplot(data=draftlottery1970,                          # layer 1, required: dataframe
       aes(x=day_birth, y=draft_number)) +             # layer 2, required: x, y definitions
       geom_point() +                                  # layer 3, geom_SOMETHING( ) required: geom_point( ) for scatter
       xlab("Day of Birth") +                          # Additional layers, not required, and as you like
       ylab("Lottery Number") +
       ggtitle("Its Hard to See a Problem")
```

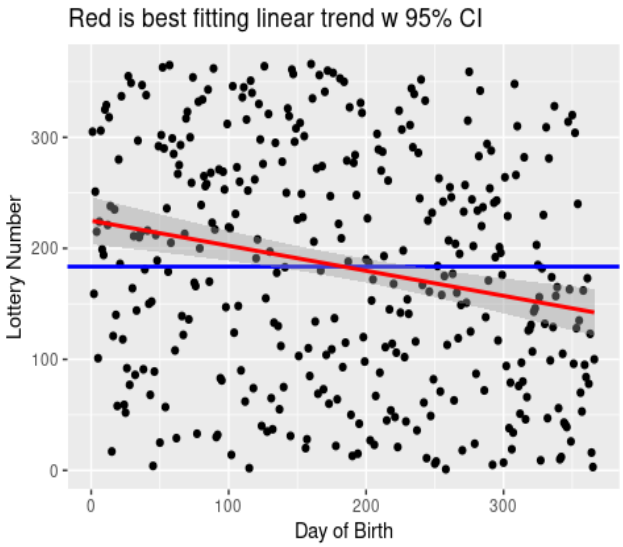
```
# add layer to show null hypothesis ("Draft lottery was Random").  Here, shown in blue.

ggplot(data=draftlottery1970,                                  # required layer: dataframe
       aes(x=day_birth, y=draft_number)) +                     # required layer: x, y definitions
       geom_point() +                                          # required layer: geom_point( ) for xy scatter

       geom_hline(yintercept=183.5, color="blue", size=1) + # added layer:  null line of random

       xlab("Day of Birth") +                                  # Add aesthetics as you like
       ylab("Lottery Number") +
       ggtitle("Blue is null (random lottery) expected trend")
```




```
# add layer to show linear fit.  Here,shown in red.
ggplot(data=draftlottery1970,                                  # required layer: dataframe
       aes(x=day_birth, y=draft_number)) +                     # required layer: x, y definitions
       geom_point() +                                          # required layer: geom_point( ) for xy scatter

       geom_hline(yintercept=183.5, color="blue", size=1) +  # added layer:  null line of random
       geom_smooth(method="lm", color="red",se=FALSE) +      # added layer:  fitted linear trend (se=FALSE to turn off CI)


       xlab("Day of Birth") +                                  # Add aesthetics as you like
       ylab("Lottery Number") +
       ggtitle("Red is best fitting linear trend")
```



```
# assessment is facilitated with display of confidence band (CI)
ggplot(data=draftlottery1970,                                  # required layer: dataframe
       aes(x=day_birth, y=draft_number)) +                     # required layer: x, y definitions
       geom_point() +                                          # required layer: geom_point( ) for xy scatter

       geom_hline(yintercept=183.5, color="blue", size=1) +    # added layer:  null line of random
       geom_smooth(method="lm", color="red") +                 # MODIFICATION HERE: restore default (so we now get CI)

       xlab("Day of Birth") +                                  # Add aesthetics as you like
       ylab("Lottery Number") +
       ggtitle("Red is best fitting linear trend w 95% CI")
```




```
analysis - simple linear regression
# Key:  lm(y ~ x, data=dataframe)
model_simple <- lm(draft_number ~ day_birth, data=draftlottery1970)        # lm( ) to fit linear model (normality assumed)

anova(model_simple)                                                        # anova( ) to show analysis of variance

## Analysis of Variance Table
##
## Response: draft_number
##            Df  Sum Sq Mean Sq F value     Pr(>F)
## day_birth   1  208098  208098  19.535 0.00001305 ***
## Residuals 364 3877529   10653
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
analysis reporting - nice looking tabular summary
library(stargazer)                                             # attach {stargazer} to access stargazer( )

stargazer(model_simple, type="text",                           # stargazer( ) to produce report of model fit.  Nice!
          report=('vc*p'),
          font.size="small",
          align=TRUE,
          title="The 1970 Draft Lottery Was Not Random")
```

```
##
## The 1970 Draft Lottery Was Not Random
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                             draft_number
## ------------------------------------------------
## day_birth                    -0.226***
##                              p = 0.00002
##
## Constant                     224.913***
##                               p = 0.000
##
## ------------------------------------------------
## Observations                    366
## R2                             0.051
## Adjusted R2                    0.048
## Residual Std. Error     103.211 (df = 364)
## F Statistic           19.535*** (df = 1; 364)
## ================================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```